Assignment 5: CH5350

Abhijeet Mavi (BE14B001)

Sample Simulations for ARMA models

```
model_aic=matrix(data=NA,nrow=3,ncol=100)
model_order=matrix(data=NA,nrow=4,ncol=100)

for (i in 1:100){
print(i)
yk=arima.sim(n=600,list(ar=c(0.2,0,0.1)))

#For AR model estimation

armod_aic={}
for (j in 1:10){

  armod=arima(yk,order = c(j,0,0))
  armod_aic[j]=armod$aic
  }
model_aic[1,i]=min(armod_aic)
model_order[1,i]=which.min(armod_aic)


# #For MA model estimation

mamod_aic={}
for (k in 1:10){

  mamod=arima(yk,order = c(0,0,k))
  mamod_aic[k]=mamod$aic
}
model_aic[2,i]=min(mamod_aic)
model_order[2,i]=which.min(armod_aic)

#For ARMA model estimation

armamod <- auto.arima(yk,seasonal = FALSE,d=0,D=0,max.p=10,max.q=10,start.p =
1,start.q = 1)

model_aic[3,i] = armamod$aic

model_order[3,i] = armamod$arma[1]

model_order[4,i] = armamod$arma[2]
}
```

```
ar3_indices = which(model_order[1,]==3)

ar3_aic<-apply(model_aic[,ar3_indices], 2, min)

count_true<-length(intersect(ar3_aic,model_aic[1,]))

print(count_true)
```

I took AR, MA and ARMA models, ranging from 0 to 10th order.

For N=600 samples, we see that we get AR(3,0) **16 times** out of 300 possible orders, for the given R script. I used **forecast** package to evaluate the arma model estimations using function **auto.arima()**.

For N=100 samples, we could only get **2 reasonable models** with AR(3,0) for AIC being the model ranker.

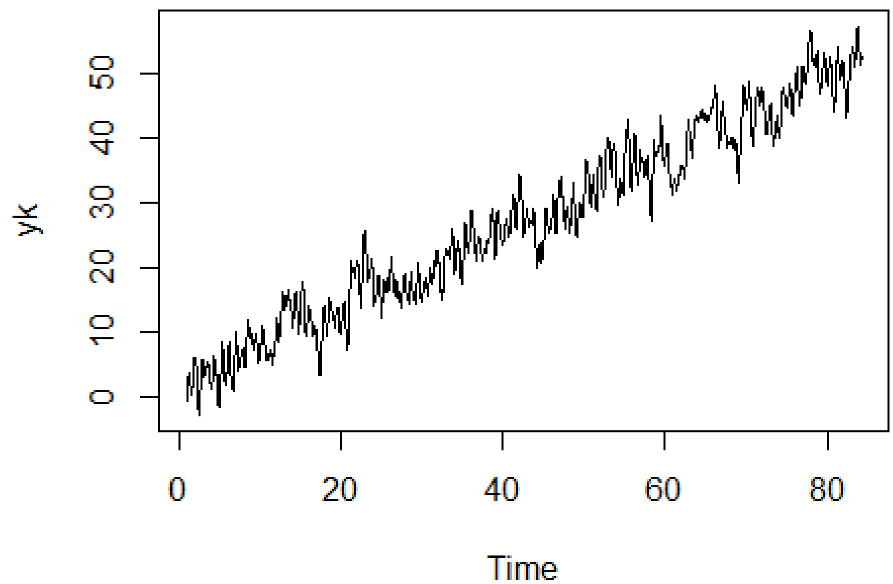
Fitting a Seasonal Model

The seasonality in a model can be evaluated in 2 ways: in respect with **additive models** or **multiplicative models**

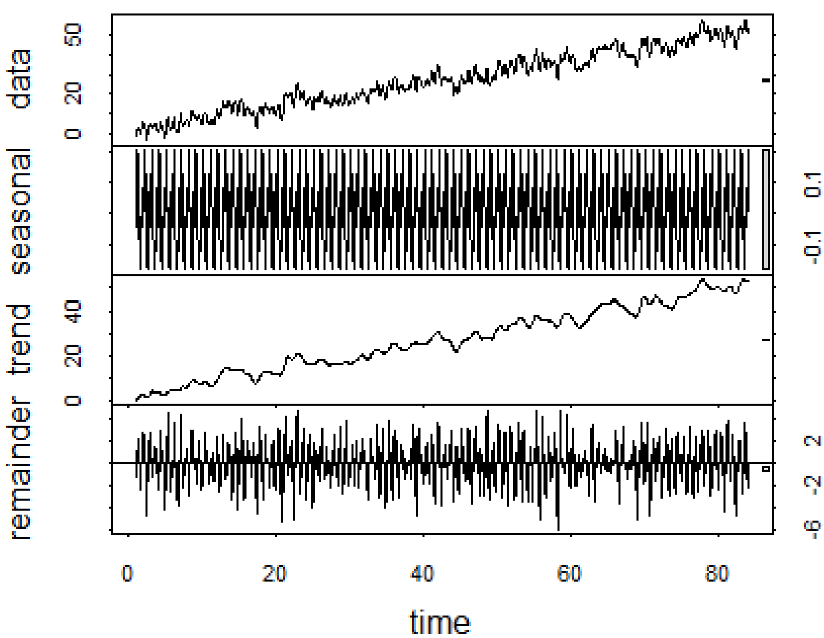
Using stl() command

We see that the series is obviously seasonal in some way. Using **stl()**, we deduce the remainder for the decomposed series.

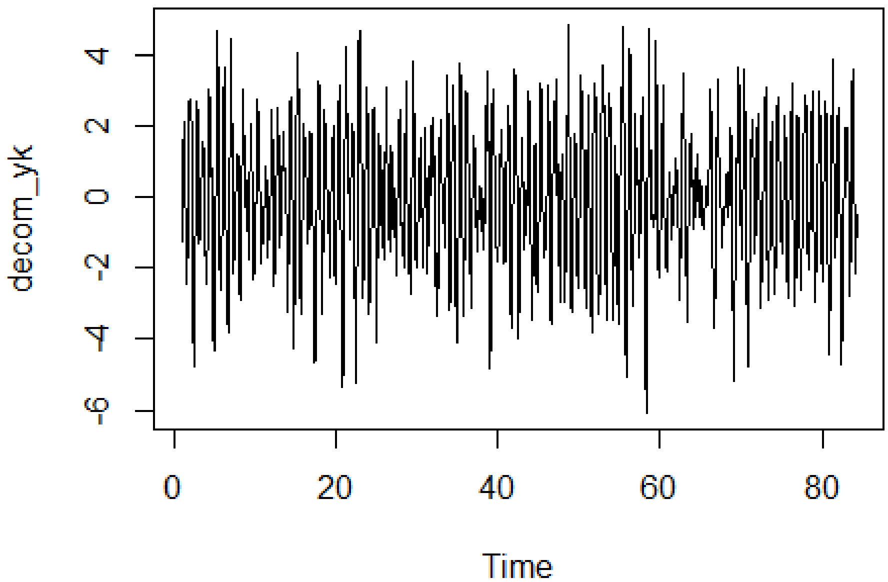
```
library('forecast')

load('sarima_data.Rdata')

#Part 1

plot(yk)
```



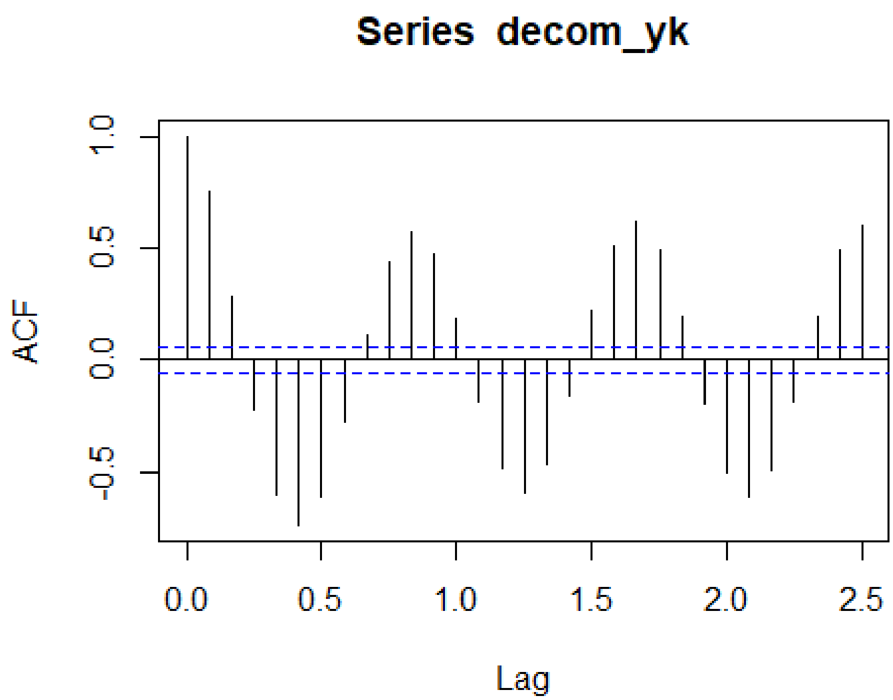

```r
plot(stl(yk,'per'))
```



```r
decom_yk <- remainder(stl(yk,'per'))

plot(decom_yk)
```

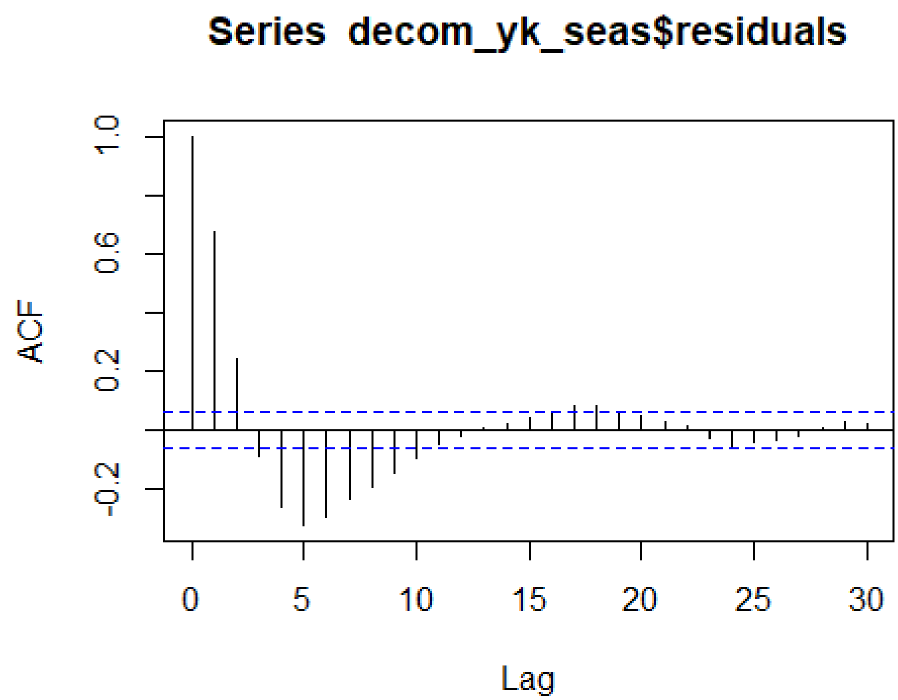



```r
acf(decom_yk)
```

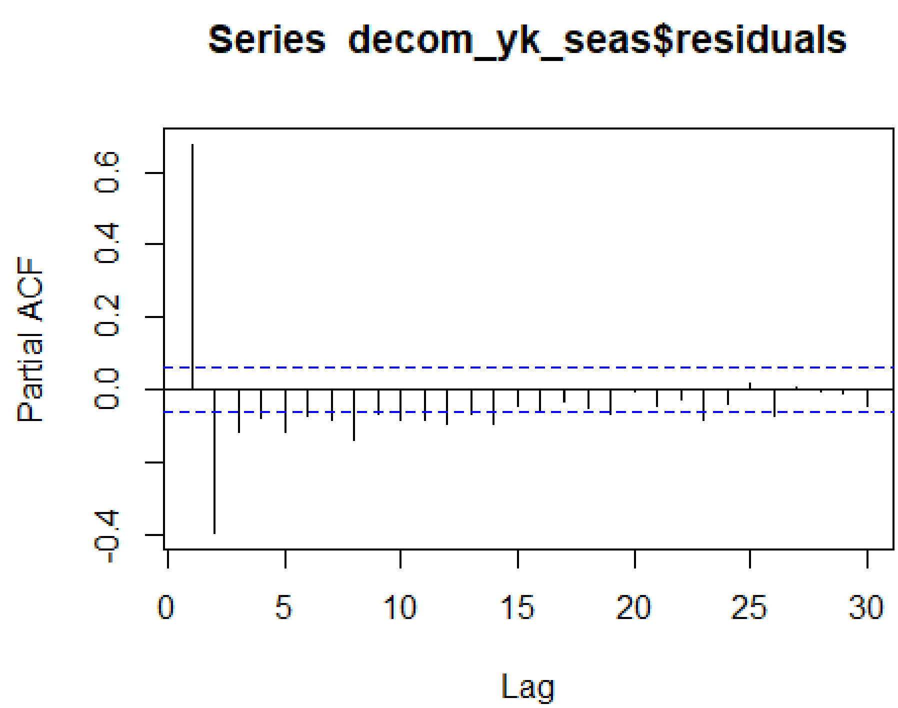


We see that the dcomposed series is seasonal with a period of about 10. We further fit a cosine wave using **lm()** and then check for its residuals.

```
tvec <- 1:1000

decom_yk_seas <- lm(decom_yk ~ I(sin(2*pi*(1/10)*tvec))+
I(cos(2*pi*(1/10)*tvec)))

acf(decom_yk_seas$residuals)
```

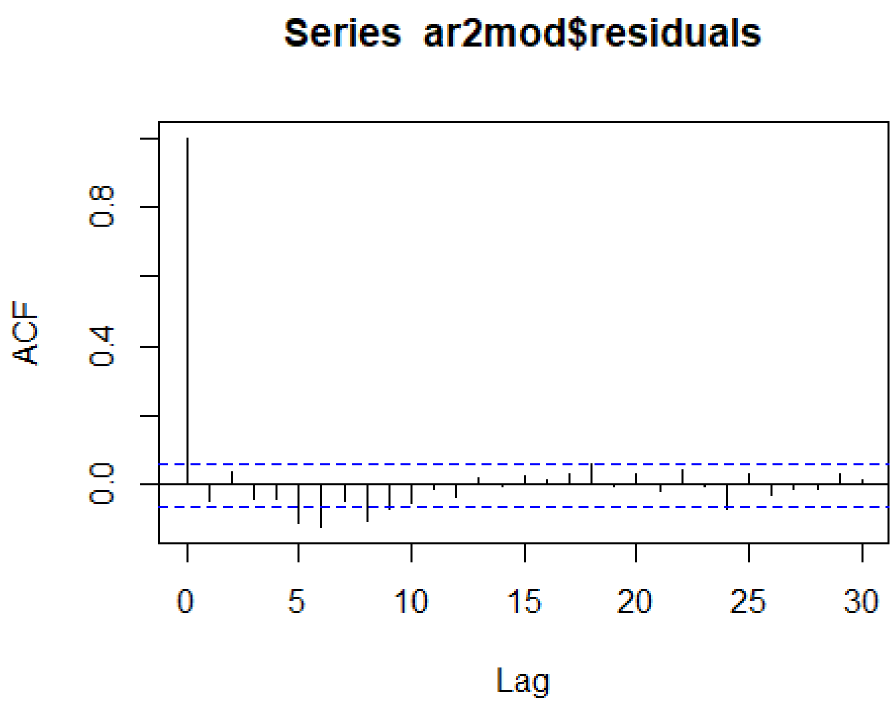



```r
pacf(decom_yk_seas$residuals)
```



The residuals clearly show that the series is AR(2) with respect to the PACF plot. Further fitting the AR(2) model

```
ar2mod <- arima(decom_yk_seas$residuals,order=c(2,0,0))

ar2mod

##
## Call:
## arima(x = decom_yk_seas$residuals, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       0.9434  -0.3940     0.0010
## s.e.  0.0290   0.0291     0.0592
##
## sigma^2 estimated as 0.7116:  log likelihood = -1249.31,  aic = 2506.63

acf(ar2mod$residuals)
```

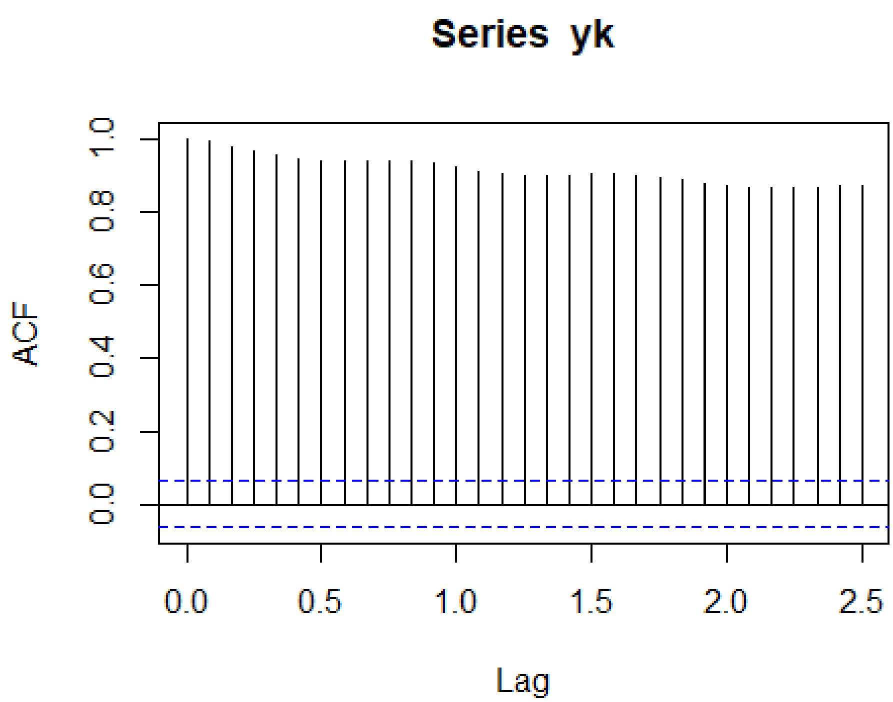


We see that the residuals of AR(2) model satisfy underfit overfit criteria with respect to the ACF of the plot. Hence, the final series is:

$$v[k] = 0.002798 sin(0.2\pi k) - 1.374654 cos(0.2\pi k) + 0.9434 * v[k-1] - 0.3940 * v[k-2] + e[k]$$

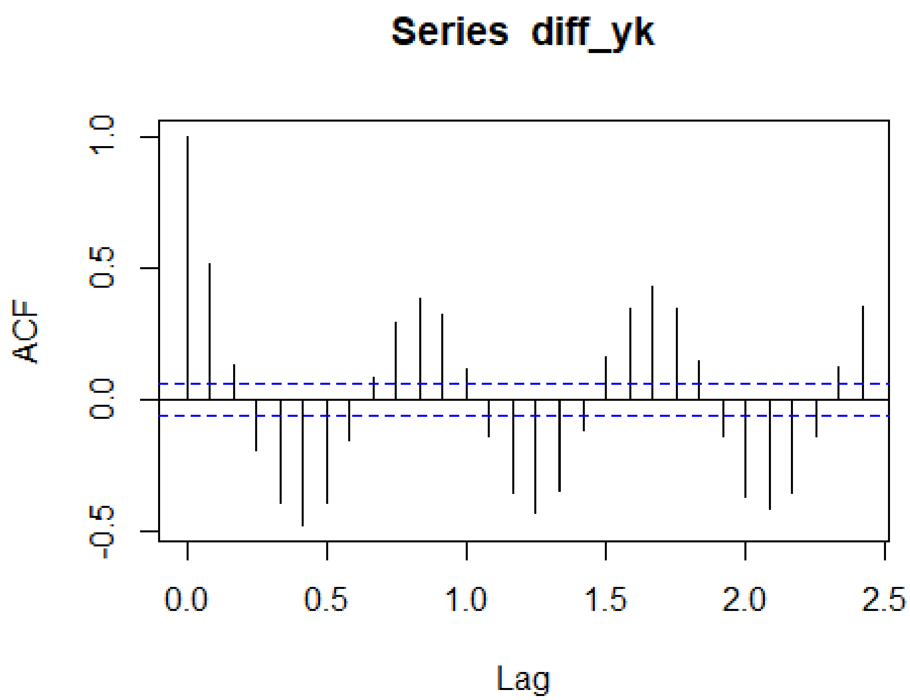
Using SARIMA Models Now,

```r
#Part 2

acf(yk)
```



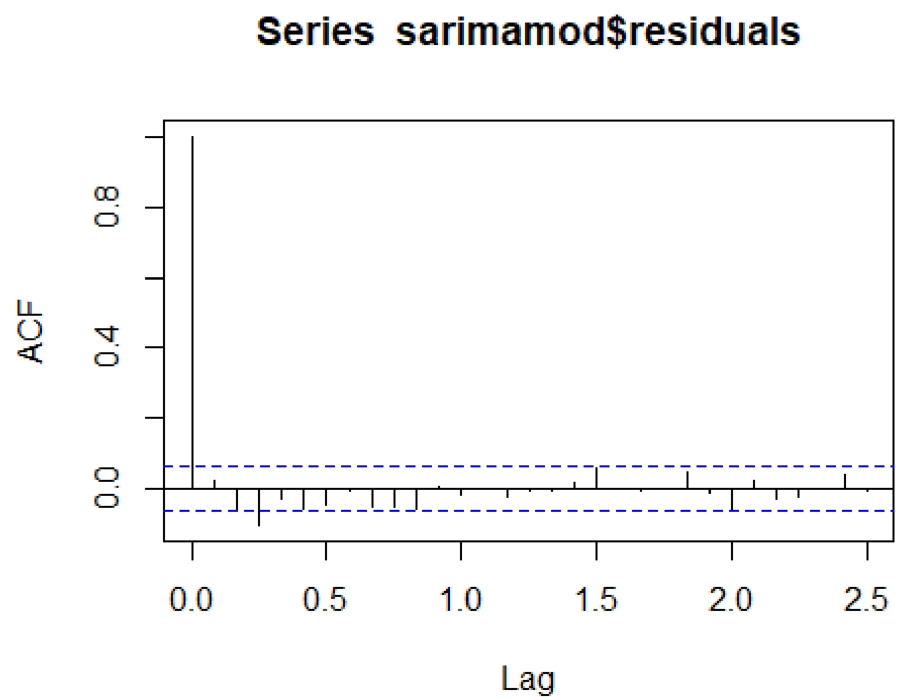
We see that the series clearly needs differencing. SO, after differencing once, we look at the acf of residuals and notice its periodicity of approximately 10.

```r
diff_yk <- diff(yk)

acf(diff_yk)
```

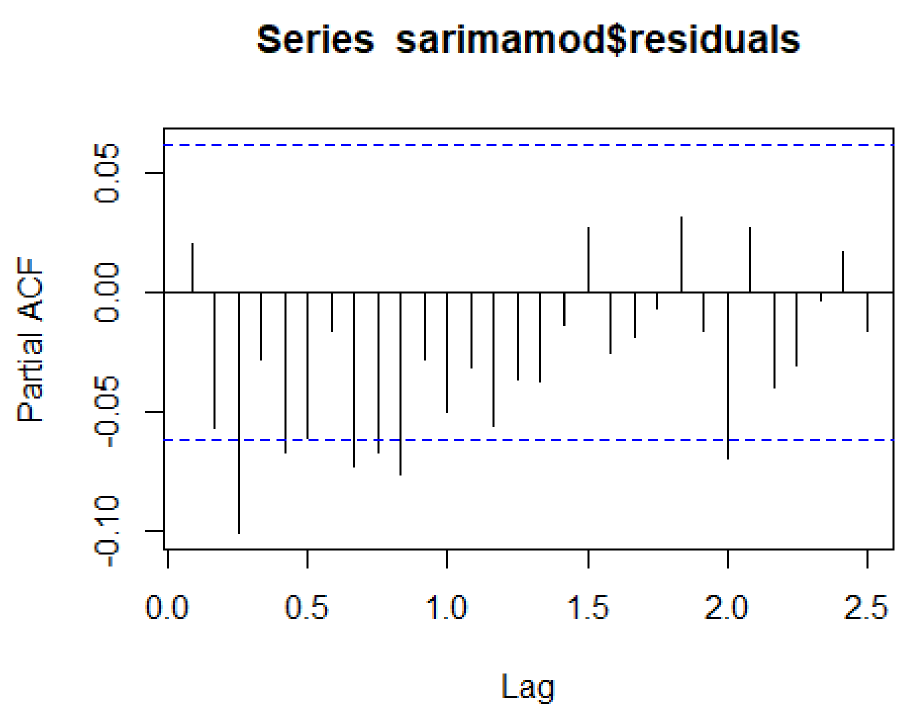


I then simulated various SARIMA models with periodicity and checked which one satisfies underfit and overfit arguements. I concluded that AR(2) with SARIMA(1,1) is the best model. The ACF and PACF plots show the same.

```
sarimamod=arima(yk,order=c(2,0,0),seasonal=list(period=10,order=c(1,0,1)))

acf(sarimamod$residuals)
```




```r
pacf(sarimamod$residuals)
```



$$\nabla v[k] = 1.3031v[k-1] - 0.3101v[k-2] + e[k]) * (0.9988v[k-10] - 0.9648e[k-10])$$

Maximum Likelihood Estimation

For a given sample set, suppose we have X_i ranging from 1 to N, with X_n, for some n being the maximum value of the set. So,

$$P(X_i|\theta) = \frac{1}{\theta}$$

Assuming the events of picking a sample is independent.

$$f(X_1, X_2, X_3 \ldots\ldots, X_N|\theta) = \frac{1}{\theta^N}$$

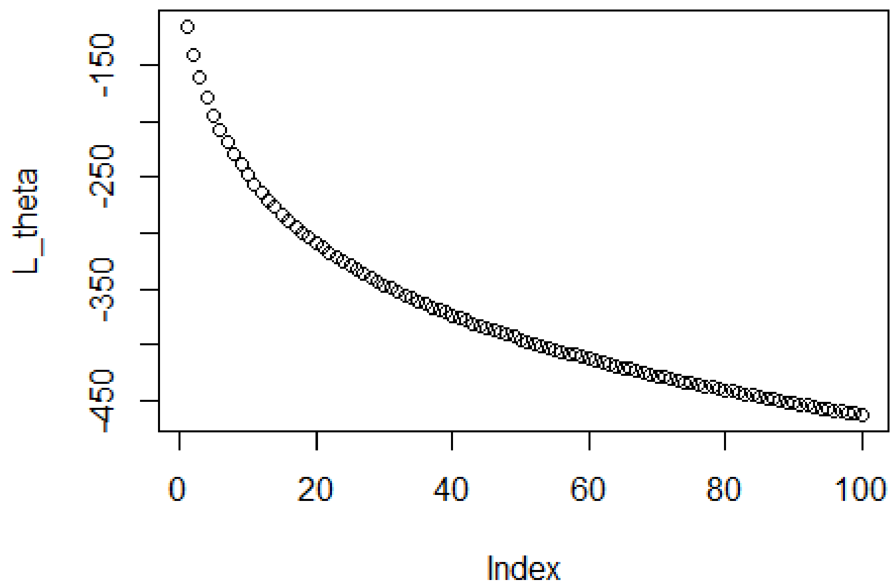
$$l(\theta|N) = \frac{1}{\theta^N}$$

$$L(\theta|N) = -N log\theta$$

NOw the log-likelihood function will take it's maximum value at the minimum value of θ. θ should atleast be greater or equal to the maximum value of sample set X_n.

$$\hat{\theta} = X_n$$

```
load('mle_unif.Rdata')

max(xk)

## [1] 1.966975

theta = seq(max(xk),101, by=1)

L_theta={}

for (i in 1:100) {

    L_theta[i] = -100*(log(theta[i]))-(sum(xk)/theta[i])

}
plot(L_theta)
```



One can deduce the same from the graph of likelihood vs θ.

Furthermore, we know that bias $\Delta\theta = E(\hat{\theta}) - \theta_0$, with θ_0 being the truth. We see that for any x in [0,θ] such that $x>= X_n, $the probability of this sample space over N samples become $P_\theta = (\frac{x}{\theta})^n$.

$$E(\hat{\theta}) = E(X_n) = \int_0^\theta P_\theta\,(X_n >= x)dx$$

$$E(X_n) = \int_0^\theta 1 - (\frac{x}{\theta})^n dx = \theta - \frac{\theta}{N+1} = \frac{N\theta}{N+1}$$

Hence,

$$Bias = \Delta\theta = E(\hat{\theta}) - \theta_0 = \frac{N\theta}{N+1} - \theta = \frac{-\theta}{N+1}$$

Rather, if the distribution was uniform over $[0,\theta)$, we would have only got a limit tending to the maximum value X_n for $\hat{\theta}$ which is not a right estimate for our subsequent bias calculation since this limit could become far away from the real X_n. In order to solve this ambiguity, one can take large number of samples, then using the principles of consistency, one can say that the limit will tend to X_n.

Fisher's Information

For a given probability distribution function $f(y|\theta)$, we evaluate likelihood $l(\theta|y)$ which can be considered equal to the the probability distribution function.

In order to estimate the most efficient estimator, the estimator should satisfy the following condition,

$$\frac{S(\theta; y)}{I(\theta)} + \theta = \theta^*$$

In the above equation $^* $should only be a function of y, that is the data instants. With that in mind, let's evaluate the two conditions given.

λ is the parameter

$$f(\lambda) = l(\lambda) = \lambda e^{-\lambda y}$$

$$L(\lambda) = log(l(\lambda)) = log(\lambda) - \lambda y$$

$$S(\lambda; y) = \frac{\partial L}{\partial \theta} = \frac{\partial(log(\lambda) - \lambda y)}{\partial \lambda} = \frac{1}{\lambda} - y$$

$$I(\lambda) = -E\left((\frac{\partial L}{\partial \theta})^2\right) = -E(\frac{-1}{\lambda^2}) = \frac{1}{\lambda^2}$$

This implies,

$$\frac{S(\theta; y)}{I(\theta)} + \theta = \lambda - \lambda^2 y + \lambda = 2\lambda - \lambda^2 y = \theta^*$$

Clearly, the most efficient estimator of λ is a function of parameter itself, hence, it is not possible to determine the efficient estimator for λ. Let's try for a modified version of the same parameter.

$\frac{1}{\lambda} = k$ is the parameter

$$f(k) = l(k) = \frac{e^{-y/k}}{k}$$

$$L(k) = log(l(k)) = \frac{-y}{k} - log(k)$$

$$S(k; y) = \frac{\partial L(k)}{\partial k} = \frac{y}{k^2} - \frac{1}{k}$$

$$I(k) = -E\left(\frac{\partial^2 L}{\partial k^2}\right) = -E\left(\frac{-2y}{k^3} + \frac{1}{k^2}\right) = \frac{1}{k^2}$$

This happens because since the process is white noise, so that makes $E(y) = \frac{1}{\lambda} = k$ since y is plotted on an exponential distribution. Now,

$$\frac{S(k;y)}{I(k)} + k = k^* = -k^2\left(\frac{1}{k} - \frac{y}{k^2}\right) + k = y$$

As one can see, k^* is purely a function of y, and hence it is the most efficient estimator for the given estimator.

Variability of Sample Mean

For the vaiability of sample mean ($\sigma^2_{\bar{y}}$), we have for stationary data that has N realisations,

$$\sigma^2_{\bar{y}} = E((\bar{y} - E(\bar{y}))^2) = E((\frac{1}{N}\Sigma_{k=0}^{N-1} y[k] - \mu_y)^2) = E((\frac{1}{N}\Sigma_{k=0}^{N-1}(y[k] - \mu_y))^2)$$

$$\sigma^2_{\bar{y}} = \frac{1}{N^2}E(\Sigma_{k=0}^{N-1}(y[k] - \mu_y))^2) + \frac{1}{N^2}E(\Sigma_{n=1}^{N}\Sigma_{m=1}^{N}(y[n] - \mu_y)(y[m] - \mu_y))$$

The first term in the above expression can be easily seen as the $\frac{\sigma[0]}{N}$, the second term where n is not equal to m can be shown as the auto-covariance function.

As, $\sigma[l] = \sigma[-l]$ for a stationary process, so it will suffice us to only take one side of the double summation, when $n < m$.

$$\sigma^2_{\bar{y}} = \frac{\sigma[0]}{N} + \frac{2}{N^2}E(\Sigma_{n=0<m}^{N-1}(y[n] - \mu_y)(y[m] - \mu_y))$$

Now, let's take $|n - m| = l$ as the lag for ACVF. For, l=1, we will see $\sigma[1]$ terms N-1 times, $\sigma[2]$ terms N-2 times and so on for $\sigma[l]$, we will see it N-l times. We can rewrite the above expression as

$$\sigma^2_{\bar{y}} = \frac{\sigma[0]}{N} + \frac{2}{N^2}E(\Sigma_{l=1}^{N-1}(N - l)\sigma[l]$$

Hence,

$$\sigma^2_{\bar{y}} = \frac{1}{N}(\sigma[0] + 2E(\Sigma_{l=1}^{N-1}(1 - \frac{l}{N})\sigma[l]$$

```
N=5000 #no. of samples

#Generic method to calculate variance of sample mean estimator

mean_yk={}

for (i in 1:N){

  yk<-arima.sim(model=list(ma=0.4,order=c(0,0,1)),n=10000)
  mean_yk[i]=mean(yk)
```

```
}

hist(mean_yk)

var_1={}
for (i in 1:5000){
  var_1[i]=(mean(mean_yk)-mean_yk[i])^2
}

var_1=sum(mean_yk)/(N-1)

#Calculation using the proved expression

acvf_data = acf(yk,lag.max = 10000, type='cov')

acvf_yk = acvf_data$acf

sum_term=0

for (i in 2:10000){

  sum_term=sum_term+((1-(i/10000))*acvf_yk[i])
}

var_2 = (1/10000)*(acvf_yk[1]+(2*sum_term))
```

The code written above simulates the given MA(1) model and I have evaluated variance with both the methods. Variance with the generic method comes out to be **4.149214e-05** and with our expression proved above, we get a variance of **1.728528e-05.** The values have a different scale altogether, but still are comparable to an extent that both are converging to 0 which is the general case for sample mean estimator. The variance would ideally converge to 0 as we increase the sample size to infinity.